

HXHYDatabase 开发文档

何轩 5060379026

一 概述

HXHYDatabase 数据库采用了多键单值映射关系的散列表作为数据结构，并使用提高外部文件访问效率的内存映射文件技术，文件类型为二进制文件。在实现原有功能的基础上作了一定的扩展，比较有特色的功能是，散列值可依据数据的规模动态生成，用户可自定义数据库的配置，使用文件大小动态扩展技术及空间整理技术，有效地兼顾了时间和空间两方面的考虑。多方面多角度的统计功能在对于性能的测试，全局的把握上起到了一定的作用。同时可将二进制文件转化为字符串流形式方便调试与直观理解数据库内部构造。在追求数据结构与算法上效率的同时，HXHYDatabase 延续了 HXHYEditor 的优点，注重用户界面的友好性与程序的健全性，努力打造出内在与外在两者和谐统一。

二 亮点介绍

★数据结构

针对本数据库特定的需求，采用了五键单值映射关系的散列表，且依据此记录长度小的特点，可以把文件与散列值直接映射。而需求中对于分数的查找为分数段形式查找，对于排序只要求依照分数进行，选择散列表无疑是明智的选择。用分数自身作为其散列值在分数的查找与排序只需依次读取并打印即可完成，而 ID，姓名又可由散列值直接定位，因此此散列表能兼顾查找与排序，在这个层面上远远优于 B+树。

★空间整理技术

为了防止散列链在删除修改中闲置记录的动态增加，采用了空间整理技术，程序可自动在闲置记录达到某特定值时进行空间整理，有效维持了散列链的长度，并可对空间进行再利用。

★用户的个性化设置

在设置菜单中，高级用户可根据数据库的具体特点设置出最高效率的配置，同时又不显烦琐，普通用户可以由软件根据用户的需要，动态生成的所需配置，使用宏与全局变量也可最大程度地增加程序的可移植性。

★对界面的完善

在设计过程中，很考虑用户对界面显示的要求，及对异常情况的处理，最大程度实现程序的健全性。

ex. 在编辑器中使用了一些特殊的符号，用于区分用户键入内容与系统输出，在指令前均有右三角形的符号标记，对于系统消息，成功提示消息使用一种笑脸，失败或警示消息使用另一种笑脸，以使用户区分。

三 模块设计

为可读性及可维护性考虑，使用了模块化的思想编程：

★控制器、显示器模块

该模块用于显示文本，以及接收用户操作并进行判断

★数据模块

该模块定义了索引及记录的数据结构

★文件模块

该模块由索引文操作与数据文件组成，并将其进行封装，外部只能通过构造函数，get，set 方法进行作

★数据库模块

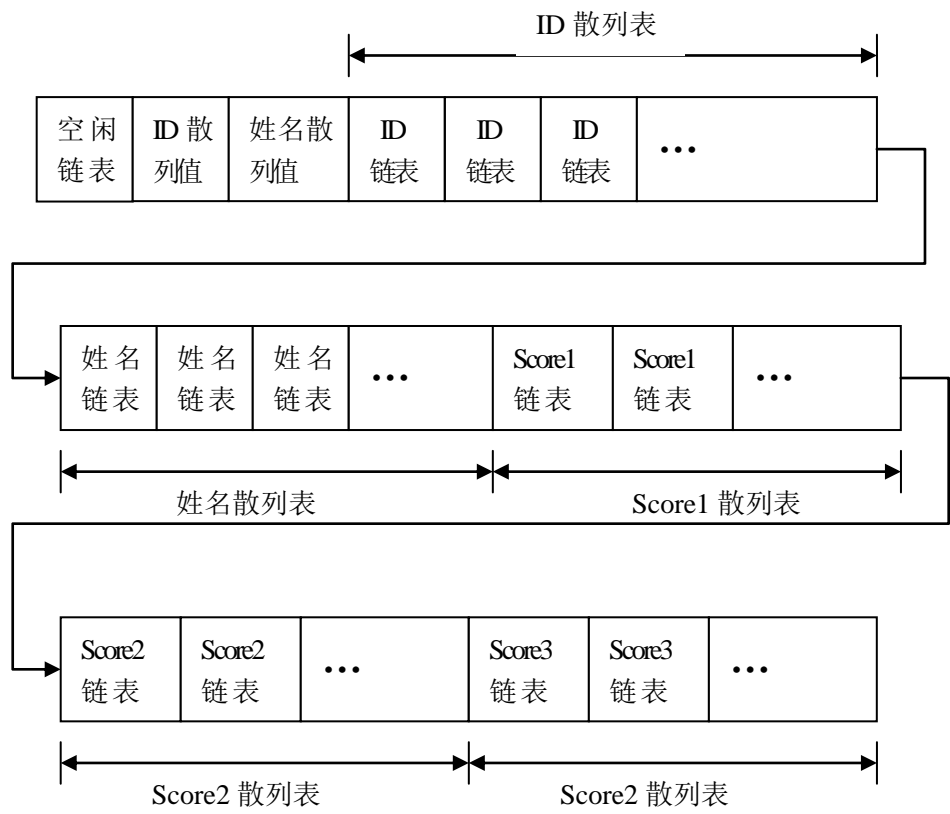
该模块即包含对数据库所进行的整个操作，包括打开，初始化，关闭，设置，添加，删除，修改，查找，排序，统计等操作

★主程序模块

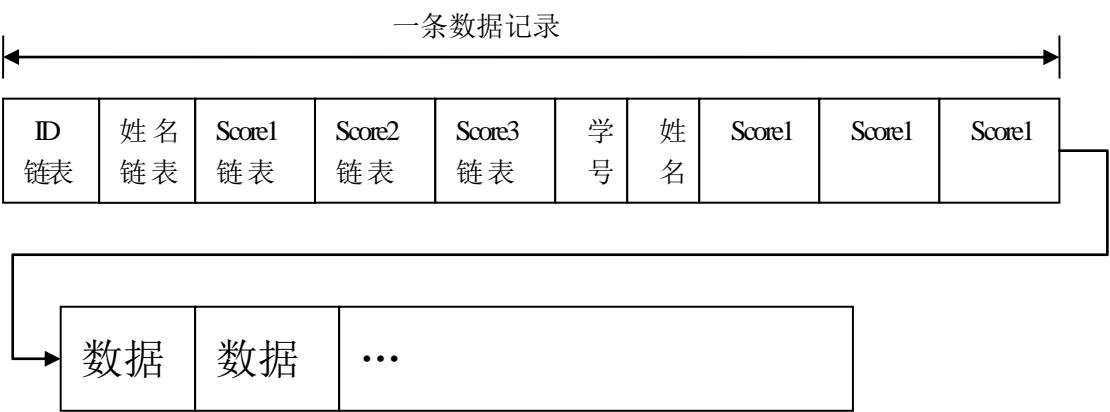
该模块控制程序的运行

四 算法与数据结构

索引文件数据结构图



数据文件数据结构图



★数据结构描述

数据结构由索引文件与数据文件组成。

索引文件主要由多键单值的散列表构成，第一个位置存放空闲链表，由于散列值动态生成，故第二第三位置记录此数据库 ID 与姓名的散列值，其后存放 ID 散列表，姓名散列表，数学成绩散列表，程序设计成绩散列表，总成绩散列表。其中 ID 与姓名散列表长度即为二者的散列值，三个成绩的散列表散列值即为分数，故长度分别为 101，101，201。

数据文件由数据记录构成，一条数据记录包括五条散列链的下一条记录的偏移量，及 ID，学号，三个成绩的值。

★算法描述

在散列算法上，对于 ID 直接采用取模法，对于姓名先采用伪十进制转换再进行取模，经测试，散列程度较高，散列值约为数据量 17 倍时最佳，数据散列程度可达 95%以上，最大链上记录条数为 3 条。且考虑所占空间数。

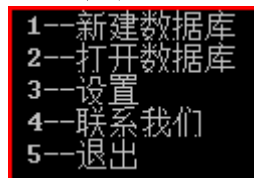
对于查找，删除，修改，排序等操作均通过操作五条散列链的值来实现。对于索引文件，长度始终不变，只修改指定位置的偏移量。对于数据文件，数据记录均依次在末尾插入（文件中有闲置可用数据除外），文件大小动态扩展，扩展大小可自定义。

空闲链表中元素为空间整理时整理出的闲置可用数据（即已删除或修改的数据）的偏移量，最后是指向数据文件末尾的偏移量。为区别两者数据，对于后者用其相反数表示（故此处为考虑偏移量为 0 的二义性，偏移量从 1 开始）

空间整理的实现即为遍历所有散列链表，将标记为空的数据从链中取下（五条链分别进行操作），并把其插入到空闲链表中（用 ID 的链表值作为空闲链表偏移量）。

五 用户手册

★主菜单



可选择 1--新建数据库 2--打开数据库 3--设置 4--联系我们 5--退出

选择新建数据库时，系统可判断该文件是否已存在

选择打开数据库时，系统可判断该文件是否存在

选择设置时可对数据库进行配置

★设置菜单

```
设置
1-----散列值
2-----文件扩展大小
3-----空间整理
4-----默认设置
5-----返回上一级
```

可选择设置 1--散列值 2--文件扩展大小 3--空间整理 4--默认设置 5--返回上一级

```
散列值设置
1----手动设置散列值
2----设置数据量自动生成
3----返回上一级
```

散列值设置中高级用户可手动设置散列值，以达到最高效率的配置

普通用户可输入数据量让后台自动生成所需的散列值

文件扩展大小设置可以根据数据量设置文件初始大小（略大于数据量为宜）及文件扩展大小此为内存映射文件技术的弊端。（动态设置可节省空间开销）

```
是否开启自动空间整理功能? Y--是 N--否
>>>> Y
请设置需要整理时的闲置数据量:
50
Ⓢ 已开启自动空间整理功能!
Ⓢ 当出现闲置数据量超过50时,将进行空间整理,且当关闭文件时自动空间整理
```

自动空间整理功能可以选择开启或关闭，并设置需要整理时的闲置数据量。开启后每达到闲置数据量即进行空间整理，当关闭数据库时，后台也会做空间整理
默认设置选项恢复成默认设置

★操作菜单

```
1--添加数据
2--删除数据
3--修改数据
4--排序
5--查询
6--随机添加数据
7--统计信息
8--空间整理
9--转换文件格式
10-关闭数据库
```

```
请输入学号:
>>>> 506037
请输入姓名:
>>>> hx
请输入数学成绩:
>>>> 99
请输入程序设计成绩:
>>>> 100
Ⓢ 所用时间: 0
Ⓢ 插入此条记录成功
```

```
请输入学号:
>>>> 506037
请输入姓名:
>>>> hx
请输入数学成绩:
>>>> 100
请输入程序设计成绩:
>>>> 100
Ⓢ 此学号已存在!
Ⓢ 所用时间: 0
```

添加数据即添加一条记录，系统先判断学号是否已存在

删除数据即删除一条记录，若未找到提示无此记录

修改数据即查找相同 ID 的记录，并对其进行修改，若未找到提示无此记录

```
1--数学成绩排序
2--程序设计成绩排序
3--总分排序
4--生成时间排序
5--返回上一级
>>>> 1
ID          name    math    program total
Ⓢ          记录条数          10000
Ⓢ 所用时间: 9
请按任意键继续. . .
```

排序中新增以生成时间排序，用以查看数据文件内部结构
（左图由于数据量太大只做排序并未打印排序结果）

1--学号查询
2--姓名查询
3--数学成绩查询
4--程序设计成绩查询
5--总分查询
6--返回上一级

请输入姓名:
▶▶▶▶ hx
3 hx 0 0 0
2 hx 98 97 195
1 hx 99 100 199
© 所用时间: 2

查询中姓名查询允许出现姓名相同的情况

请输入插入随机记录的数量:
▶▶▶▶▶ 10000
© 所用时间: 134
© 随机插入成功

输入随机记录数量即可随机插入数据，学号随机数是由两个四位时间随机数拼接而成的 8 位数，姓名为 8 个小写字母拼接而成，每个小写字母均用时间随机数产生，两分数用时间随机

数直接生成

the hash num of id is: 9717
the hash num of name is: 9721
the max chain of id is: 4
the max chain of name is: 3
the max chain of s1 is: 127
the max chain of s2 is: 124
the max chain of s3 is: 112
the total number of data is: 10000
the number of idle data is: 0
the number of waste data is: 0
the hash number of id is: 177511
the hash number of name is: 177511
initial size of data file is: 10100
the expand unit is: 100
the waste num to arrange is: 100
auto arrange function is: open
© 所用时间: 71

统计信息中，系统将对数据库进行全面的统计，从而可了解当前数据库的技术指标，运行状况，效率情况，以及配置。统计信息包括 ID，姓名已用链数，五条链的最大链长，数据量，闲置数据量，可用数据量，ID，姓名散列值，文件初始大小，文件扩展大小，自动整理开闭情况及整理所需闲置数据量




只是统计多数是遍历进行，故较别的操作较耗时间(左图为 71ms)

手动空间整理完成!

选择空间整理即可手动进行空间整理，整理完成后闲置数据量为 0

转换为文本文件成功!

选择转换文件格式即可将索引与数据文件从二进制文件转换为字符串流的文本格式

 hx.hx	2007/12/27 20:47	HX 文件	1,389 KB
 hx.hy	2007/12/27 20:47	HY 文件	1 KB
 hx-idx	2007/12/27 21:48	文本文档	1,042 KB
 hx-dat	2007/12/27 21:48	文本文档	1 KB

五 时间空间复杂度分析

首先申明一下，虽然采用了内存映射文件技术，但内部实现是通过偏移量直接定位，故并不是把整个文件读入内存中，用流操作同样可以实现，使用此技术只是为了加快硬盘数据访问与写入速度。

★空间复杂度分析

散列值的动态生成，及文件大小动态扩展，保证了空间的最小开销。在此若将长整型，字符串，整型视为同样的内存开销 a ，并设数据量 n ，散列值分别为 ID_HSZ，NAME_HSZ，101，101，201，则实际数据所占空间为 $5an$ ，索引文件所占空间为 $a(3+ID_HSZ+NAME_HSZ+403) \approx a(ID_HSZ+NAME_HSZ)$ (此两链的散列值通常很大)，数据文件所占空间为 $10an$ ，故总空间为 $a(10n+ID_HSZ+NAME_HSZ)$ ，若按照默认设置，即散列值为数据量的 9 倍，有总空间为 $28an$ ，实际数据占总数据的 18% 左右，可见此数据结构还是消耗了大量的空间

★时间复杂度分析

因为内存映射文件极大的提高了数据读取与写入速度，故运行效率是非常高的。这里假设每次访问的时间为 a ，写入时间为 b ，而相对于读取数据，内存进行的一些操作，如赋值运算，方法调用的时间可忽略不计。在此设数据量为 n ，默认散列值通常可保证 95% 左右的散列程度，平均散列链在 1 左右，最大散列链在 3 左右，由此可估算出：

添加平均时间： $8a+7b$	添加最大时间： $10a+7b$
删除平均时间： $2a+b$	删除最大时间： $4a+b$
修改平均时间： $10a+8b$	修改最大时间： $14a+8b$
学号查询平均时间： $2a$	学号查询最大时间： $4a$
姓名查询平均时间： $2a$	姓名查询最大时间： $4a$
成绩查询平均时间： $(x+50)a$	成绩查询最大时间： $(x+101)a$ (x 为数量)
总成绩查询平均时间： $(x+100)a$	总成绩查询最大时间： $(x+201)a$ (x 为数量)
成绩排序平均时间： $(n+101)a$	成绩排序最大时间： $(n+101)a$ (假设全覆盖)
总成绩排序平均时间： $(n+201)a$	总成绩排序最大时间： $(n+201)a$
生成时间排序平均时间： na	生成时间排序最大时间： na

六 性能测试与探究

对数据库各个方面的性能进行实际的测试与探究：

★散列值与散列程度的关系

选取 10000 条记录作为测试标准，测试散列值与散列程度的关系

散列值	ID链数	最大链数	平均链数	姓名链数	最大链数	平均链数	倍数
348709	9842	2	1.016054	9856	3	1.01461	34.8709
177511	9724	4	1.028383	9721	4	1.028701	17.7511
136189	9605	3	1.041124	9632	4	1.038206	13.6189
93701	9508	4	1.051746	9515	3	1.050972	9.3701
68207	9328	3	1.072041	9285	4	1.077006	6.8207
42689	8968	4	1.115076	8920	4	1.121076	4.2689
25693	8335	4	1.19976	8290	5	1.206273	2.5693
17189	7629	6	1.310788	7583	6	1.318739	1.7189

通过数据可知，综合考虑占用空间大小与散列程度，最大链数，平均链数，取散列值为数据量 9 倍左右为易

★同一散列算法下数据量与散列程度的关系

在依照数据量动态生成的散列值下，数据量与 ID 链，姓名链，ID 及姓名最大链间的关系：

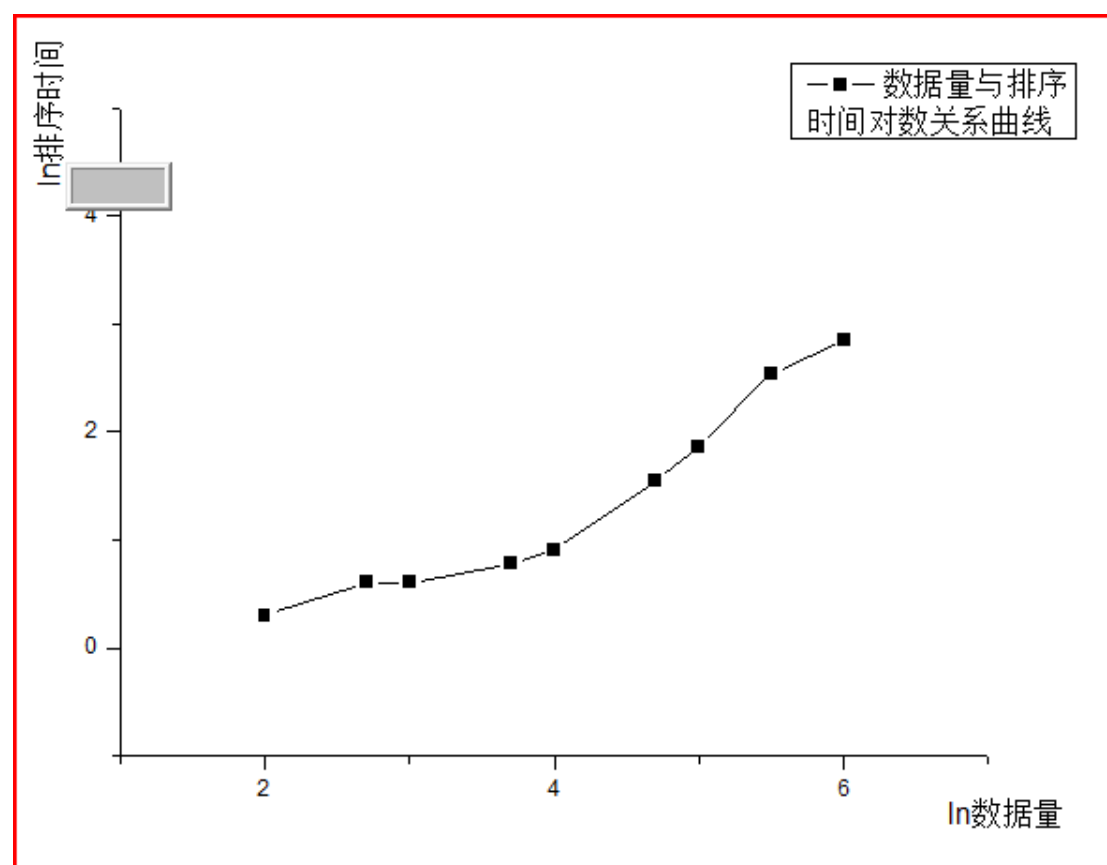
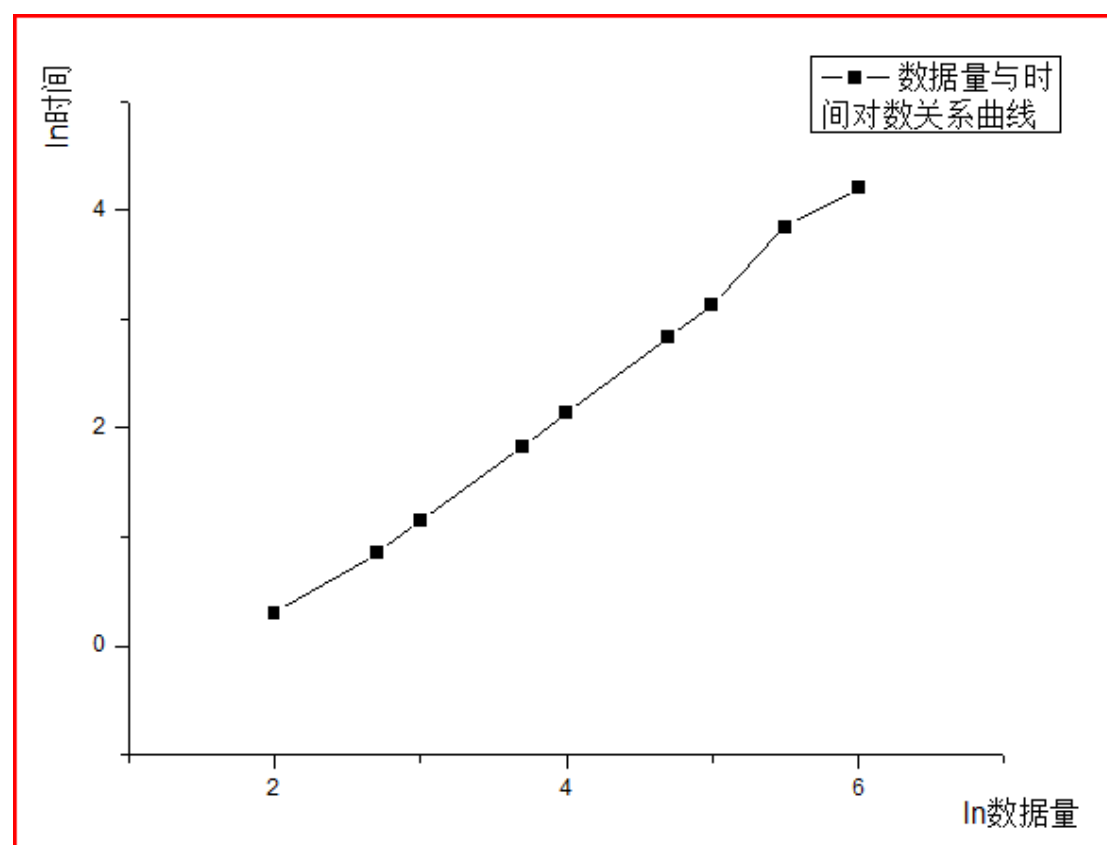
数据量	散列值	ID链	姓名链	ID最大链	姓名最大链	散列程度
100	5501	99	98	2	2	0.985
500	9103	489	484	2	2	0.973
1000	13613	973	970	3	3	0.9715
5000	49603	4747	4774	3	4	0.9521
10000	94603	9493	9502	4	4	0.94975
50000	454603	48687	48611	3	4	0.97298
100000	904601	94729	94682	4	4	0.947055
500000	4504601	474222	472998	4	4	0.94722
1000000	9004603	951068	945370	5	5	0.948219

通过数据可知，此散列算法（9 倍数据量）基本不受数据量的影响，保持了很高的散列程度，高达 95%以上

★同一散列算法下数据量与时间的关系

在依照数据量动态生成的散列值下，数据量及生成时间，插入时间，删除时间，查找时间，排序时间的关系：

数据量	散列值	生成时间	插入时间	删除时间	查找时间	排序时间
100	5501	2	0	1	1	2
500	9103	7	0	1	1	4
1000	13613	14	0	1	1	4
5000	49603	66	0	1	1	6
10000	94603	137	0	1	1	8
50000	454603	677	0	1	1	35
100000	904601	1338	0	1	1	72
500000	4504601	7053	0	1	1	349
1000000	9004603	15758	0	1	1	695



通过数据及生成时间，排序时间与数据量的对数曲线图可知，此散列算法下，生成时间于数据量大致成线性关系，排序时间也近似成线性趋势。而对于插入，删除，修改（插入+删除）时间则几乎随数据量变化，而且效率非常高。

七 一些思考

★对数据库需求的理解

需求是针对此特定的数据类型及需求来构建的以追求效率为主的数据库，还是以此例子为踏板来构建一个普遍适用的以追求通用性为主的数据库？

若为前者，此数据库在学号查找上，排序上均可以优于其他数据结构。因为成绩的散链表是以分数本身作为散列值的，但由于链表很长，故若追求通用性的话，此数据库不太适用

★ 判断数据库优劣的标准

以哪些标准为主？

☆数据结构与算法

☆时间利用率

☆空间利用率

☆可维护性安全性

☆可移植性

☆人性化智能化设计

☆健全性

何 轩

2007-12-28